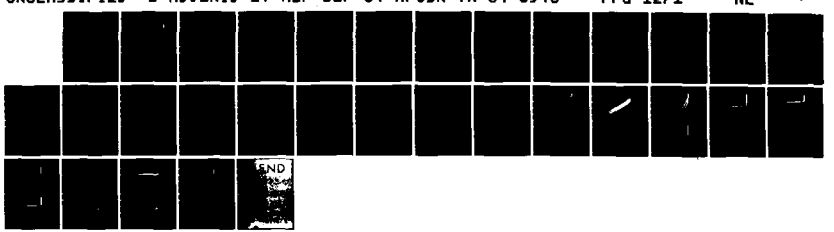
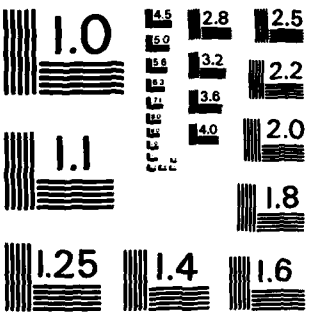
AD-A147 719 A MOVING FINITE ELEMENT METHOD FOR TIME DEPENDENT PARTIAL DIFFERENTIAL EQ..(U) RENSSELAER POLYTECHNIC INST TROY NY DEPT OF MATHEMATICAL SCIE.. 1/1

UNCLASSIFIED S ADJERID ET AL. SEP 84 AFOSR-TR-84-0946 F/G 12/1 NL





A MOVING FINITE ELEMENT METHOD FOR TIME DEPENDENT PARTIAL DIFFERENTIAL EQUATIONS WITH ERROR ESTIMATION AND REFINEMENT*

Slimane Adjerid** and Joseph E. Flaherty***

Dedicated in memory of Richard C. DiPrima

Abstract

Sept 84

We discuss a moving finite element method for solving vector systems of time dependent partial differential equations in one space dimension. The mesh is moved so as to equidistribute the spatial component of the discretization error in H^1. We present a method of estimating this error by using p-hierarchic finite elements. The error estimate is also used in an adaptive mesh refinement procedure to give an algorithm that combines mesh movement and refinement.

We discretize the partial differential equations in space using a Galerkin procedure with piecewise linear elements to approximate the solution and quadratic elements to estimate the error. A system of ordinary differential equations for mesh velocities are used to control element motions. We use existing software for stiff ordinary differential equations for the temporal integration of the solution, the error estimate, and the mesh motion. Computational results using a code based on our method are presented for several examples.

*The authors were partially supported by the U. S. Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant Number AFOSR 80-0192 and the U. S. Army Research Office under Contract Number DAAG29-82-K-0197. The first author also received support from the government of Algeria.

**Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12181. This work will be used in partial fulfillment of his Ph.D. requirements at the Rensselaer Polytechnic Institute.

***Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12181.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S): AFOSR-TR- 84_0946
6a NAME OF PERFORMING ORGANIZATION: Rensselaer Polytechnic Institute; 6b. OFFICE SYMBOL (If applicable)	7a NAME OF MONITORING ORGANIZATION: Air Force Office of Scientific Research
6c. ADDRESS (City, State and ZIP Code): Department of Mathematical Sciences, Troy NY 12181	7b ADDRESS (City, State and ZIP Code): Directorate of Mathematical and Information Sciences, Bolling AFB DC 20332
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: AFOSR; 8b. OFFICE SYMBOL (If applicable): NM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: AFOSR-80-0192
8c ADDRESS (City, State and ZIP Code): Bolling AFB DC 20332	10 SOURCE OF FUNDING NOS: PROGRAM ELEMENT NO: 61102F; PROJECT NO: 2304; TASK NO: A3; WORK UNIT NO:

11 TITLE (Include Security Classification)
A MOVING FINITE ELEMENT METHOD FOR TIME DEPENDENT PARTIAL DIFFERENTIAL EQUATIONS WITH /ERROR ESTIMATION AND REFINEMENT

12 PERSONAL AUTHOR(S)
Slimane Adjerid and Joseph E. Flaherty*

13a TYPE OF REPORT	13b. TIME COVERED	14 DATE OF REPORT (Yr., Mo., Day)	15 PAGE COUNT
Technical	FROM _____ TO _____	SEP 84	27

16 SUPPLEMENTARY NOTATION
*Dept of Computer Science, RPI, Troy NY 12181.

17 COSATI CODES		
FIELD	GROUP	SUB GR

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Moving finite element method; adaptive methods; time dependent partial differential equations; error estimation; refinement.

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The authors discuss a moving finite element method for solving vector systems of time dependent partial differential equations in one space dimension. The mesh is moved so as to equidistribute the spatial component of the discretization error in H^1. They present a method of estimating this error by using p-hierarchic finite elements. The error estimate is also used in an adaptive mesh refinement procedure to give an algorithm that combines mesh movement and refinement.

The authors discretize the partial differential equations in space using a Galerkin procedure with piecewise linear elements to approximate the solution and quadratic elements to estimate the error. A system of ordinary differential equations for mesh (CONTINUED)

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS ☐	UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL: CPT John F. Thomas, Jr.	22b TELEPHONE NUMBER (Include Area Code): 5026

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE

84 11 16 022

ITEM #19, ABSTRACT, CONTINUED: velocities are used to control element motions. The authors use existing software for stiff ordinary differential equations for the temporal integration of the solution, the error estimate, and the mesh motion. Computational results using a code based on this method are presented for several examples.

1. Introduction

Many technological situations involve the rapid formation, evolution, propagation, and disintegration of small scale structures. Some examples are shock waves, shear layers in laminar and turbulent flows, phase boundaries during nonequilibrium thermal processes, and classical boundary layers. With increasing complexity of the physical problem, there is an increasing need for reliable and robust software tools to accurately and efficiently describe the phenomena. Adaptive techniques automatically change and evolve with the solution and are thus good candidates for providing the computational methods and codes necessary to solve some of these difficult problems.

Two types of adaptive techniques are currently popular: (i) moving mesh methods, where a grid of a fixed number of finite difference cells or finite elements is moved so as to follow and resolve local nonuniformities in the solution, and (ii) local refinement methods, where uniform fine grids are added to coarser grids in regions where the solution is not adequately resolved. A representative sample of both types of methods is contained in Babuska, Chandra, and Flaherty [1]. Roughly speaking, moving mesh methods are superior at reducing dispersive errors in the vicinity of wave fronts while local refinement methods can, in principle, add enough fine grids to resolve any fine scale structure (cf. Hedstrom and Rodrigue [9]).

We discuss a moving mesh finite element procedure for finding numerical solutions of m-dimensional vector systems of partial differential equations having the form

$$(1.1) \qquad Lu := M(x,t)u_t + f(x,t,u,u_x) - [D(x,t,u)u_x]_x = 0,$$

$$0 < x < 1, \; t > 0,$$

subject to the initial and boundary conditions

(1.2) $$u(x,0) = u^0(x) , \quad 0 \le x \le 1 ,$$

(1.2b,c) either $u_i(x,t) = a_i(t)$ or $\sum_{j=1}^{m} D_{ij}u_{j_x}(x,t) = a_i(t)$

at $x = 0$ and 1 , $t > 0$, $i = 1, 2, \ldots, m$.

We are primarily concerned with parabolic problems where D and M are positive definite; however, we do not restrict ourselves to this case, but instead we assume that conditions are specified so that equations (1.1,2) have an isolated solution.

We discretize (1.1,2) in space using a finite element-Galerkin procedure with piecewise linear approximations on a moving mesh. We simultaneously calculate an error estimation using a piecewise quadratic correction. This error estimate is used to move the mesh so that it approximately equidistributes the local spatial component of the discretization error. Temporal integration is performed using a code for stiff ordinary differential equations and algebraic systems due to Petzold [15]. We halt the temporal integration at specified times and examine our error estimate. If it is larger than a prescribed tolerance, the step is rejected and the integration is re-done using a finer spatial discretization. On the other hand, if the error estimate indicates that the solution is being calculated too accuarately, then the integration is continued with a coarser spatial mesh.

Our procedure differs from the moving finite element method of Miller et al. [8,12,13] in that we move the mesh so that the spatial error in H^1 is equidistributed and they move their mesh so as to minimize the residual in L_2. Our refinement procedure also differs from local refinement procedures of, e.g., Berger [2], Bieterman and Babuska [3,4], and Flaherty and Moore [6]. Our mesh equidistributes the local error and, thus, when refinement is necessary it is performed globally. This avoids the use of complicated tree

data structures that are necessary with the local refinement schemes of, e.g., Berger [2] and Flaherty and Moore [6,7].

In Section 2 of this paper we discuss our discretization procedure, in Section 3 we describe our algorithm and mesh refinement techniques, in Section 4 we apply a code based on our algorithm to several linear and nonlinear examples, and in Section 5 we discuss our results and suggest some future considerations and refinements.

2. Discrete Formulation

We construct a weak form of (1.1,2) in the usual manner. Thus, we assume $u \in H^1_E$, where the subscript E denotes that u satisfies any essential (Dirichlet) boundary conditions in (1.2). We select a test function $v \in H^1_0$, where the subscript 0 indicates that v satisfies homogeneous versions of any essential boundary conditions. We multiply (1.1) by v, integrate it on $0 \leq x \leq 1$, and then integrate the diffusive terms by parts to obtain

$$(2.1a) \qquad (v,u_t) + (v,f) + a(v,u) = 0 \ , \quad \text{for all } v \in H^1_0 \ , \ t > 0 \ ,$$

where

$$(2.1b) \qquad (v,u) = \int_0^1 v(x,t)^T u(x,t)dx \ ,$$

$$(2.1c) \qquad a(v,u) = \int_0^1 v_x^T D(x,t,u)u_x dx - v^T D(x,t,u)u_x \Big|_0^1 \ .$$

To construct finite element solutions of (2.1) we select finite dimensional approximations $U \in S^N_E$ and $V \in S^N_0$ to u and v, respectively. Here, S^N_E and S^N_0 are finite dimensional subspaces of H^1_E and H^1_0, respectively. Thus, we seek to find $U \in S^N_E$ satisfying

(2.2a) $(V,U_t) + (V,f) + a(V,U) = 0$, for all $V \in S_0^N$, $t > 0$,

(2.2b) $(V,U) = (V,u^0)$, $t = 0$.

We introduce a partition

(2.3) $\Pi(t,N) := \{0 = x_0(t) < x_1(t) < \ldots < x_N(t) = 1\}$

of (0,1) into N subintervals $(x_{i-1}(t),x_i(t))$, $i = 1, 2, \ldots, N$, $t > 0$ and select U and V to be piecewise linear polynomials with respect to this partition. Thus, for example, U has the form

(2.4a) $$U(x,t) = \sum_{i=0}^{N} U_i(t)\phi_i(x,\Pi(t,N)) ,$$

where

(2.4b) $$\phi_i(x,\Pi(t,N)) = \begin{cases} \dfrac{x - x_{i-1}(t)}{x_i(t) - x_{i-1}(t)} , & x_{i-1}(t) \leq x \leq x_i(t) \\ \dfrac{x_{i+1}(t) - x}{x_{i+1}(t) - x_i(t)} , & x_i(t) \leq x \leq x_{i+1}(t) \\ 0 , \text{ otherwise} \end{cases}$$

We note that the derivative U_t in (2.2a) has the form

(2.5) $$U_t = \sum_{i=0}^{N} \{\dot{U}_i(t)\phi_i(x,\Pi(t,N)) + \sum_{j=i-1}^{i+1} U_i(t) \frac{d\phi_i}{dx_j} \dot{x}_j(t)\} ,$$

where $\dot{(\ )} := d(\ )/dt$. Thus, if the mesh positions and velocities, $x_i(t)$ and $\dot{x}_i(t)$ were known, we would have a set of ordinary differential equations for the nodal values $U_i(t)$ of $U(x,t)$.

In order to estimate the error in the piecewise linear finite element equations (2.2) to (2.5) we write

$$u(x,t) = U(x,t) + e(x,t) \tag{2.6}$$

and substitue this into (2.1a) to obtain

$$(v,U_t + e_t) + (v,f(.,t,U+e,U_x + e_x)) + a(v,U+e) = 0 \ , \tag{2.7}$$

$$\text{for all } v \in H_0^1 \ , \ t > 0 \ .$$

We select a finite dimensional approximation $E \in \hat{S}_E^N$ of e consisting of piecewise hierarchic quadratic functions, i.e.,

$$E(x,t) = \sum_{i=1}^{N} E_i(t)\Psi_i(x,\Pi(t,N)) \ , \tag{2.8a}$$

where

$$\Psi_i(x,\Pi(t,N)) = \begin{cases} \dfrac{4[x - x_{i-1}(t)][x_i(t) - x]}{[x_i(t) - x_{i-1}(t)]^2} \ , \\ \qquad\qquad x_{i-1}(t) < x < x_i(t) \ . \\ 0 \ , \qquad\qquad \text{otherwise} \end{cases} \tag{2.8b}$$

We also select a finite dimensional approximation $V \in \hat{S}_0^N$ of v consisting of piecewise quadratics. Thus, we solve

$$(V,U_t + E_t) + (V,f(.,t,U+E,U_x + E_x)) + a(V,U+E) = 0 \ , \tag{2.9a}$$

$$\text{for all } V \in \hat{S}_0^N \ , \ t > 0 \ ,$$

subject to the initial conditions

$$(V,E) = (V,u^0-U) \ , \text{ for all } V \in \hat{S}_0^N \ , \ t = 0 \ . \tag{2.9b}$$

Once again, if the mesh positions and velocities were known, we would have a set of ordinary differential equations for $E_i(t)$. Note that our error estimate $E(x,t)$ assumes the superconvergence of the piecewise linear finite element solution $U(x,t)$, i.e., we assume that the piecewise linear solution is converging to higher order at the nodal points $x_i(t)$, $i = 0, 1, \ldots, N$.

The error estimate that is calculated by solving (2.9) is used to control the motion of the mesh. Specifically, we determine the mesh positions by solving the ordinary differential system

$$\dot{x}_i(t) - \dot{x}_{i-1}(t) = -\lambda(|| E_i \Psi_i ||_1 - \bar{E}) \ , \ i = 1, 2, \ldots N, \tag{2.10a}$$

where λ is a positive constant and

$$\overline{NE}^2 = || E ||_1^2 := \int_0^1 [(E)^2 + (E_x)^2] dx \ . \tag{2.10b}$$

Thus, $||E_i \Psi_i||_1$ is the local error in H^1 on $(x_{i-1}(t), x_i(t))$ and $\bar{E}$ is the average error in H^1. If $||E_i \Psi_i||_1$ is larger than $\bar{E}$ then the right hand hand side of (2.10a) is negative and the points $x_i(t)$ and $x_{i-1}(t)$ are moved closer together. Similarly, when $||E_i \Psi_i||_1$ is smaller than $\bar{E}$ the points $x_i(t)$ and $x_{i-1}(t)$ are moved apart. The differential system (2.10) was studied by Coyle et al. [5] and shown to asymptotically equidistribute the quantity $||E_i \Psi_i||_1$ and to be stable relative to small perturbations in the mesh positions. Large values of the parameter λ give shorter relaxation times of $x_i(t)$, $i = 0, 1, \ldots, N$, to an equidistributing mesh; however, they introduce stiffness into the system (2.10). This strategy is similar to one suggested by Hyman and Naughton [10] and we, like they, solve it by eliminating $\bar{E}$ using (2.10) on two neighboring intervals. This gives

$$(2.11)\qquad \dot{x}_{i+1} - 2\dot{x}_i + \dot{x}_{i-1} = -\lambda(||E_{i+1}\Psi_{i+1}||_1 - ||E_i\Psi_i||_1) ,$$
$$i = 1, 2, \ldots, N-1 , \; t > 0 .$$

We select an initial mesh for (2.11) that is either uniform or that equidistributes E(x,0). An algorithm for calculating an equidistributing mesh is described in, e.g., Coyle et al. [5].

At present, we solve the three sets of ordinary differential equations (2.2), (2.9), and (2.11), respectively, for the piecewise linear approximation, the quadratic error estimate, and the mesh positions simulataneously using the backward difference code DDASSL developed by Petzold [15]. This code is capable of solving systems of implicit differential and algebraic equations having the general form

$$(2.12)\qquad g(t, y(t), \dot{y}(t)) = 0 , \; t > 0 ,$$

and subject to appropriate initial conditions. It can solve stiff systems, with variable order of accuracy and temporal step size, and can even solve problems when the matrix M in (1.1) is singular. Our differential system is simpler than (2.12) and has the general form

$$(2.13)\qquad A(t,y)\dot{y} + h(t,y) = 0 , \; t > 0 .$$

Furthermore, the matrix A and the Jacobian of h are sparse and banded and DDASSL is capable of exploiting this structure. At present, we use finite difference formulae to approximate Jacobians of g with respect to y and $\dot{y}$.

3. Mesh Refinement Algorithm

A top level algorithm for solving the system (2.2), (2.9), and (2.11) is presented in Figure 1 in a pseudo-PASCAL language. The input parameters are a

set of output times toutput[k], k = 0, 1, ..., K, a spatial error tolerance tol, and a value for λ.

```
PROCEDURE mferef(toutput, K, tol, λ);
  BEGIN
    select initial mesh;
    k := 1;
    WHILE k ≤ K DO
      BEGIN
        solve the problem on toutput[k-1] to touput[k] using
          the moving finite element procedure described in
          Section 2;

        compute Ē;

        IF Ē > tol
          THEN add a mesh point to each subinterval
          ELSE
            BEGIN

              IF Ē < tol/10 THEN delete a mesh point from
                                 each subinterval;
              k := k + 1
            END
      END
  END;
```

Figure 1. Top level Algorithm for the moving finite element procedure with mesh refinement.

We halt the temporal integration at every output point and examine the average error $\bar{E}$. If $\bar{E}$ is larger than tol, we add a finite element uniformly to each subinterval and re-do the integration. If $\bar{E} < tol/10$ we delete every other element and continue the integration. In the case when $tol/10 \leq \bar{E} \leq tol$ we continue the integration with the same number of elements.

Initial conditions are needed whenever elements have been added or deleted. When refinement is necessary, we calculate a refined solution $U_r(x,t)$ by interpolating U + E at time t using (2.4a) and (2.8a), i.e.,

$$(3.1a) \qquad U_r(x,t) = \sum_{i=0}^{N} U_i(t)\phi_i(x,\Pi(t,N)) + \sum_{i=1}^{N} E_i(t)\,\Psi_i(x,\Pi(t,N)) .$$

If one nodal point is added to each subinterval, then we need values of $U_r(x_i,t)$, $i = 0, 1, \ldots, N$ and $U_r((x_i + x_{i+1})/2,t)$, $i = 0, 1, \ldots, N-1$. We also need a refined error estimate $E_r(x,t)$ and we calculate this as

$$(3.1b) \qquad E_r(x,t) = \sum_{i=1}^{N} E_i(t)\,\Psi_i(x,\Pi(t,N)) - \sum_{i=1}^{N} E_i(t)\phi_{2i-1}(x,\Pi(t,2N)) .$$

Again, if one nodal point is added to each subinterval, then we need values of $E_r((x_i+x_{i+1})/4,t)$ and $E_r(3(x_i+x_{i+1})/4,t)$, $i = 1, 2, \ldots, N$.

When mesh points are deleted, we need values of E on the coarser mesh. If every other mesh point is deleted then we use

$$(3.2) \qquad E(x_{2i-1},t) = U(x_{2i-1},t) - [U(x_{2i},t) + U(x_{2i-2},t)]/2, \qquad i = 1, 2, \ldots, N-1.$$

Initial values for $U_i(0)$ and $E_i(0)$ were obtained by interpolating the exact initial function $u^0(x)$. The initial mesh $\Pi(0, N)$ can be selected so as to equidistribute $E(x,0)$. Finally, we select a constant value of λ in a relatively ad hoc manner; however, we are studying ways of automatically choosing this parameter.

Procedures for evaluating $f(x,t,u,u_x)$, $M(x,t)$, $D(x,t,u)$, and the initial and boundary conditions must also be provided. A temporal error tolerance and other parameters that are required by the code DDASSL are set internally in a relatively problem independent manner.

4. Examples

We used a code that is based on the algorithm of Section 3 to solve the following three examples that illustrate the performance of the moving finite

element method and the utility of the error estimation strategy. In order to estimate convergence rates, we solved some of the examples without refinement and on stationary meshes.

Example 1. We consider the simple linear heat conduction problem

$$(4.1) \qquad u_t + u_x - u_{xx} = f(x,t) \ , \ t > 0 \quad 0 < x < 1.0 \ ,$$

and select the source term $f(x,t)$, the initial function $u^0(x)$, and Dirichlet boundary conditions so that the exact solution of (4.1) is

$$(4.2) \qquad u(x,t) = \{1 - \tanh[C_1(x-C_2t-C_3)]\}/2 \ .$$

The solution (4.2) is a traveling wave and its steepness, speed, and phase can be determined by selecting C_1, C_2, and C_3, respectively.

We solved this problem on stationary and moving meshes with a fixed number of elements. In each case we chose $C_1 = 10$, $C_2 = 1$, $C_3 = -0.15$, and, for the moving mesh, $\lambda = 10$. Our results for the exact error $||e(.,t)||_1$, and the effectivity index

$$(4.3) \qquad \theta = ||E(.,t)||_1 \ / ||e(.,t)||_1$$

at $t = 0.5$ are presented as functions of the number of elements N in Table 1. The mesh trajectories for $N = 20$ are shown for $0 < t < 1.4$ in Figure 2. An examination of Table 1 shows that $||e||_1 = O(1/N)$ for both fixed and moving meshes, which is as expected [16]. The error of the moving mesh solution is about half of the error of the fixed mesh solution when $N < 40$. Furthermore, the effectivity index $\theta \to 1$ as N increases and this indicates that the error estimate E converges to the true error in H^1 . The mesh is concentrated in the vicinity of the wave front and follows the wave with approximately the correct speed.

We next solve this problem on a moving mesh with refinement. We select an error tolerance of 0.1, $\lambda = 30$, and show the solution $U(x,t)$ and the mesh trajectories in Figures 3 and 4, respectively. Elements are added as the pulse enters the region $0 \leq x \leq 1$ for small t and then they are deleted as the pulse leaves the region when t is about 1.25.

Example 2. We consider the following problem for Burgers' equation:

$$\begin{aligned} &u_t + uu_x - \varepsilon u_{xx} = 0 , \quad t > 0 , \quad 0 \leq x \leq 1 . \\ &u(x,0) = u^0(x) , \quad u(0,t) = a(t) , \; u(1,t) = b(t) . \end{aligned} \tag{4.4}$$

As a first example, we select u^0, a, and b so that the exact solution of (4.4) is the travleling wave

$$u(x,t) = 1 - 2\sqrt{\varepsilon}\tanh[(x-t)/\sqrt{\varepsilon}] . \tag{4.5}$$

As in Example 1, we solved this problem on stationary and moving meshes with a fixed number of elements. We chose $\varepsilon = 0.01$ and, for the moving mesh, $\lambda = 10$. Our results for the exact error $||e(.,t)||_1$, and the effectivity index at $t = 0.5$ are presented as functions of the number of elements N in Table 2. We see that $||e||_1 = O(1/N)$ for both fixed and moving meshes and that the error estimate converges to the true error. Here, as in Example 1, we see that the effectivity index is converging at a faster rate for the uniform mesh than for the moving mesh.

As a second example for (4.4), we select

$$u^0(x) = 10x(x - 1)(x - 3/4) , \; a(t) = b(t) = 0 . \tag{4.6}$$

For small times and ε, the exact solution is a pulse that moves in the positive x direction while steepening. At about $t = 1$, a shock layer forms near $x = 1$ and after a time of $O(1/\varepsilon)$, the solution dissipates to zero.

STATIONARY MESH			MOVING MESH	
N	$\|\|e\|\|_1$	θ	$\|\|e\|\|_1$	θ
10	0.4275	0.9634	0.2810	0.9232
20	0.2330	0.9941	0.1604	0.9874
40	0.1175	0.9985	0.0892	0.9978

TABLE 1. Exact error $||e||_1$ and effectively index θ for Example 1. Calculations well performed on stationary and moving meshes having N elements.

STATIONARY MESH			MOVING MESH	
N	$\|\|e\|\|_1$	θ	$\|\|e\|\|_1$	θ
10	0.2036	0.8027	0.1499	0.7321
20	0.09372	0.9724	0.0847	0.9144
40	0.04707	0.9950	0.0434	0.9854

TABLE 2. Exact error $||e||_1$ and effectively index θ for Example 2 with the exact solution given by eq. (4.5). Calculations were performed on stationary and moving meshes with N elements.

We present the solution U(x,t) and the mesh trajectories for computations performed with 20 moving elements, $\varepsilon = 0.01$, and $\lambda = 10$ in Figures 5 and 6, respectively. The mesh is concentrated in regions of high curvature and follows the moving wave front.

Example 3. We consider the reaction-diffusion model in one dimension (cf. Kapila [11])

$$u_t = u_{xx} - Due^{-\delta/T}, \qquad t > 0, \ 0 < x < 1. \tag{4.7a,b}$$

$$LT_t = T_{xx} + \alpha Du\, e^{-\delta/T},$$

$$D = R\frac{e^{\delta}}{\alpha\delta} \tag{4.7c}$$

$$u(x,0) = T(x,0) = 1, \tag{4.7d,e}$$

$$u_x(0,t) = T_x(0,t) = 0, \tag{4.7f,g}$$

$$u(1,t) = T(1,t) = 1. \tag{4.7h,i}$$

This model discribes a single one step reaction (A → B) of a gas in a region $0 < x < 1$. The quantity u is the mass fraction of the reacting gas, T is the gas temperature, L is the Lewis number, α is the heat release, δ is the activation energy, D is the Damkohler number, and $R > 0.88$ is the reaction rate.

We first consider the case when $L = 1$. Then $T + \alpha u = 1 + \alpha$ and (4.7) reduces to the simpler problem

$$T_t = T_{xx} + D(1+\alpha-T)e^{-\delta/T}, \quad t > 0, \ 0 < x < 1, \tag{4.8a}$$

$$T_x(0,t) = 0, \ T(1,t) = 1, \ t > 0, \tag{4.8b,c}$$

$$T(x,0) = 1, \quad 0 < x < 1. \tag{4.8d}$$

For small times the temperature gradually increases from unity with a "hot spot" forming at $x = 0$. At a finite time, ignition occurs and the temperature at $x = 0$ jumps rapidly from near unity to $1 + \alpha$. A sharp flame front then forms and propagates towards $x = 1$ with speed proportional to $e^{\alpha\delta/2(1+\alpha)}$. In real problems, α is about unity and δ is large; thus, the flame front moves exponentially fast after ignition. The problem reaches a steady state once the flame propagates to $x = 1$.

We solve (4.7c,8) for $\alpha = 1$, $\delta = 20$, and $R = 5$ using a moving mesh with 20 elements. Our results for the mesh trajectories when $\lambda = 10$ and 200 are shown in Figures 7 and 8, respectively. The computed temperatures T vs. x are shown in Figure 9 for several times and $\lambda = 10$ and 200. We also computed solutions on a moving mesh with refinement using tol = 0.1 and $\lambda = 300$. The mesh trajectories and solution in this case are shown in Figures 10 and 11, respectively.

We see that the ignition time and the computed solutions while the flame front is propagating from $x = 0$ to $x = 1$ are sensitive to the choice of λ (cf. Figures 9 and 11). This is due to the extremely fast rate of ignition and velocity of the front. All three solutions are in essential agreement before ignition and upon approaching steady state. This is a very difficult problem and yet our methods are capable of finding a solution with relative ease. This is because the mesh is moving with approximately the speed of the front and the solution along the mesh trajectories is changing slowly.

As a final problem, we solve the coupled system (4.7) with $L = 0.9$. As in the previous example, we take $\alpha = 1$, $\delta = 20$, $R = 5$ and solve the problem on a moving mesh with 20 elements. Our results for the mesh trajectories when $\lambda = 10$ and 150 are shown in Figures 12 and 13, respectively. The computed

temperature and mass fraction are shown in Figure 14 for $\lambda = 10$ and in Figure 15 for $\lambda = 150$. Again we computed a solution with refinement using tol = 0.1 and $\lambda = 500$. The mesh trajectories and solution are shown in Figures 16 and 17, respectively. Once again, the ignition time and computed solutions have different phases while the flame front is moving. We show the estimated error $||E||_1$ as a function of time for solutions calculated with $\lambda = 10$, N = 20, $\lambda = 150$, N = 20, and $\lambda = 500$ with refinement in Figure 18. For N = 20, the estimated error is about 50 percent smaller when $\lambda = 150$ than for $\lambda = 10$. The estimated error is less than the prescribed tolerance of 0.1 when refinement is used.

5. Discussion of Results

We have presented a moving finite element method where the mesh is moved so as to equidistribute the spatial component of the discretization error in H^1. We also discuss a computationally simple and effective procedure for estimating the discretization error using quadratic hierarchic finite elements. The error estimate gives users some confidence in the computed solution and we also use it to develop a simple procedure for combining mesh movement and refinement. Our approach makes use of existing software for solving ordinary differential equations. We have used the differential algebraic system solver DDASSL [15]; however, there are several other possibilities. We have applied a code based on our method to several examples and have shown that it can effectively solve some very difficult partial differential systems with little or no user intervention.

While our results are very encouraging, there are several aspects of our work that are still incomplete. For some problems (e.g., Example 3) the

computed solutions seem to be sensitive to the choice of the parameter λ used in the equidistribution scheme. So far, the choice of this parameter is left to the user, and while it is our only problem dependent parameter, we hope to be able to determine it adaptively in the future. Furthermore, our refinement procedure is not optimal and needs further study. However, even though our strategy was simple it was still effective in that computer times were often much less for greater accuracy with refinement than for solutions computed with a fixed number of elements.

We intend to extend our method in several directions, such as including higher order finite element approximations and to higher spatial dimensional problems. Two-and three-dimensional problems are quite difficult; however, we note that some success was reported by Miller [14] with a different, but related, moving finite element method.

References

[1] I. Babuska, J. Chandra, and J. E. Flaherty, (Eds.), Adaptive computational methods for partial differential equations, SIAM, Philadelphia, 1983.

[2] M. J. Berger, Adaptive mesh refinement for hyperbolic partial differential equations, report No. STAN-CS-82-924, Department of Computer Science, Stanford University, 1982.

[3] M. Bieterman and I. Babuska, The finite element method for parabolic equations. I A posteriori error estimation, Numer. Math., 40, 1983, pp 339-371.

[4] M. Bieterman, and I. Babuska, The finite element method for parabolic equations.II A posteriori error estimation and adaptive approach, Numer. Math., 40, 1982, pp 373-406.

[5] J. M. Coyle, J. E. Flaherty, and R. Ludwig, On the stability of mesh equidistribution strategies for time dependent partial differential equations, submitted to J. Comp. Phys., 1984.

[6] J. E. Flaherty and P. K. Moore, An adaptive local refinement finite element method for parabolic partial differential equations, Int. Conf. Accuracy Estimates and Adaptive Finite Element Computations, Int. Assn. Compu. Mech., 2, Lisbon, 1984, pp. 139-152.

[7] J. E. Flaherty and P. K. Moore, A local refinement finite element method for time dependent partial differential equations, to appear in Trans. 2nd Army Conference of Applied Math. and Comp., R.P.I., May 1984.

[8] R. J. Gelinas, S. K. Doss and K. Miller, The moving finite element method: application to general partial differential equations with multiple large gradients, J. Comp. Phys., 40, 1981, pp. 202-249.

[9] G. Hedsrtom and G. H. Rodrigue, Adaptive numerical methods for time dependent partial differential equations, UCRL-87242 preprint, Lawrence Livermore National Laboratory, Livermore. Also to appear in Proc. Conf. on Multigrid Methods, Cologne, November 1981.

[10] J. M. Hyman and M. J. Naughton,Static rezone methods for tensor-product grids, to appear in Proc. SIAM-AMS Conference on Large Scale Computation in Fluid Mechanics, SIAM, Philadelphia, 1984.

[11] A. K. Kapila, Asymptotic treatement of chemically reacting systems, Pitam Applicable Mathematics Series, 1983.

[12] K. Miller and R. N. Miller, Moving finite element. I, SIAM J. Num. Anal., 18 (1981), pp. 1019-1032.

[13] K. Miller, Moving finite element. II, SIAM J. Num. Anal., 18 (1981), pp. 1033-1057.

References (Continued)

[14] K. Miller, Moving node finite element method, to appear in Accuracy Estimates and Adaptivity for Finite Element, I. Babuska, O. C. Zienkiewicz and E. Arantes e Oliveira (Eds.), John Wiley, London, 1985.

[15] L. R. Petzold, A description of DDASSL: a differential/algebraic system solver, Sandia Report No. Sand 82-8637, Sandia National Laboratory, Livermore, 1982.

[16] G. Strang and G. Fix , An analysis of the finite element method, Prentice Hall, Englewood Cliffs, 1973.

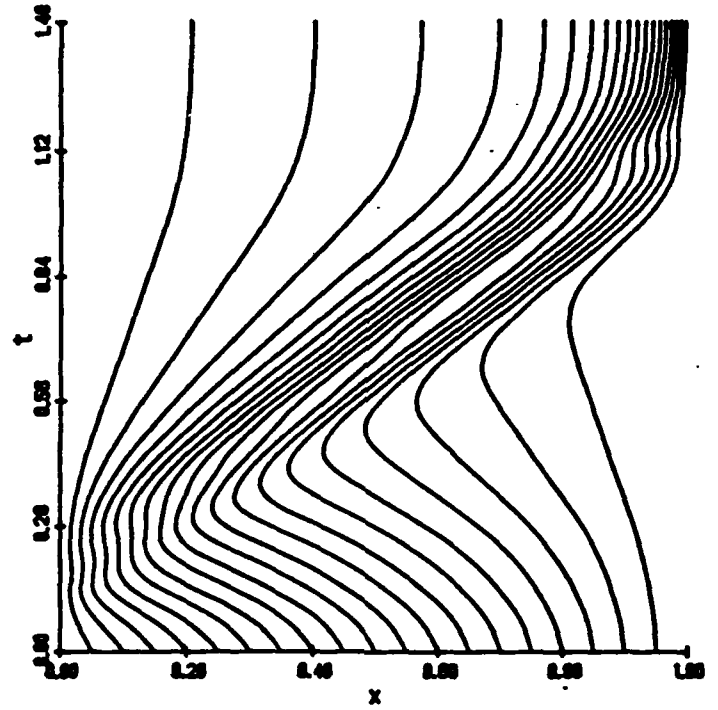


Figure 2. Mesh trajectories for Example 1 with λ=10 and N=20.

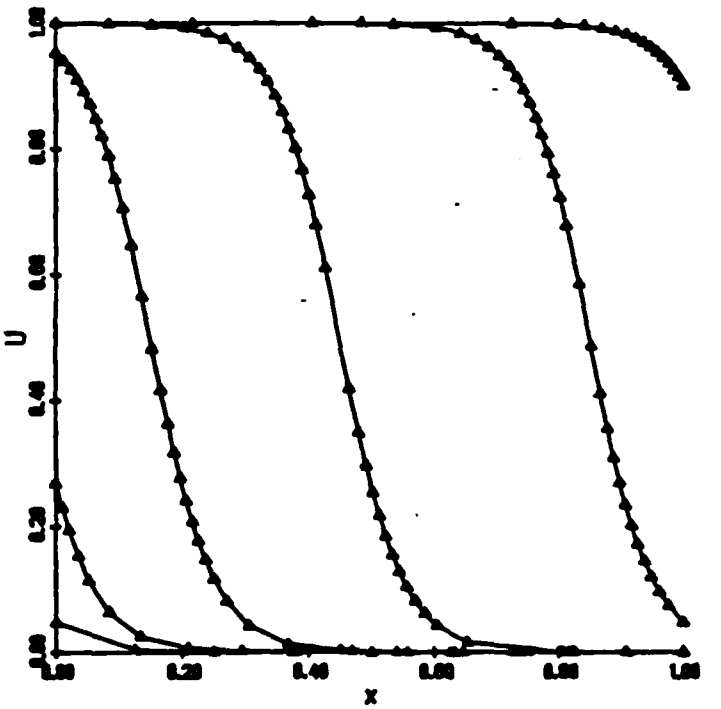


Figure 3. Solution U(x,t) vs. x at t = 0.0, 0.1, 0.3, 0.6, 1 and 1.26 for Example 1 using refinement with tol = 0.1 and λ = 30.

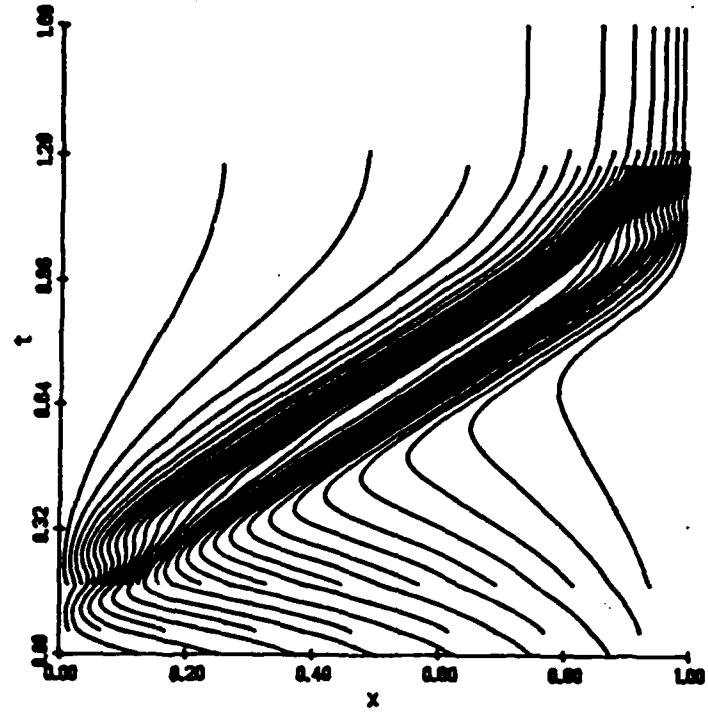


Figure 4. Mesh Trajectories for Example 1 using Refinement with tol = 0.1 and λ = 30.

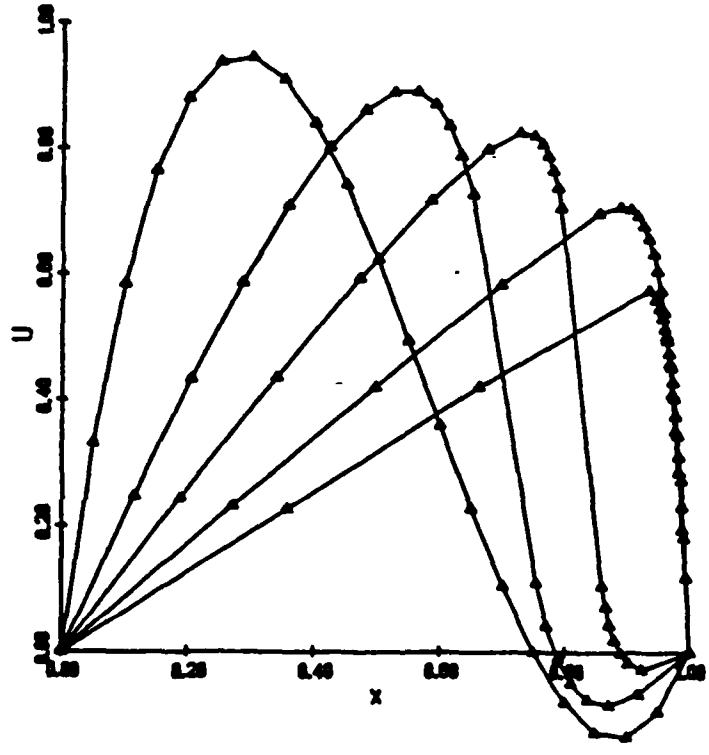


Figure 5. Solution U(x,t) vs. x for t = 0.0, 0.3, 0.6, 1.0 and 1.4 for Example 2. Initial and boundary conditions are chosen to satisfy Eq, (4.6), λ = 10, N = 20 and ε = 0.01.

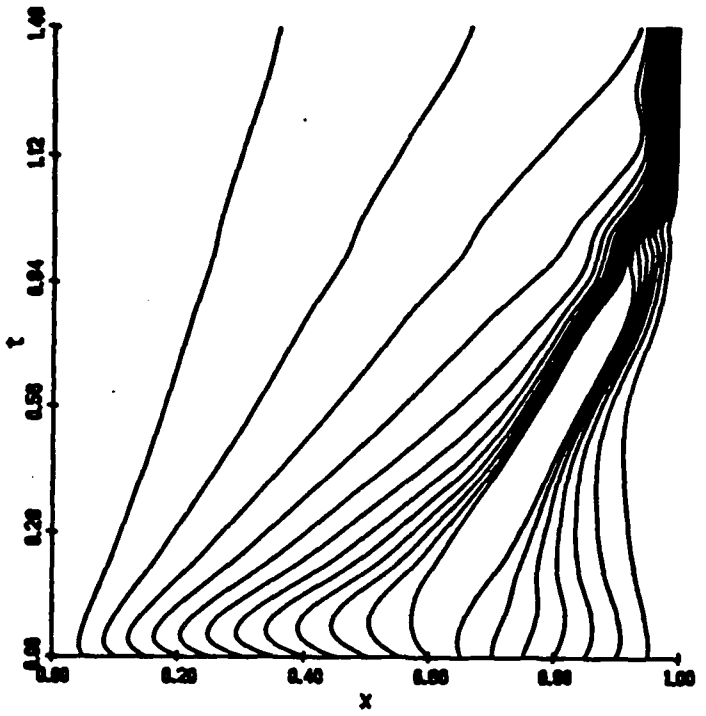


Figure 6. Mesh trajectories for Example 2. Initial and boundary conditions are chosen to satisfy Eq. (4.6), $\lambda = 10$, $N = 20$ and $\varepsilon = 0.01$.

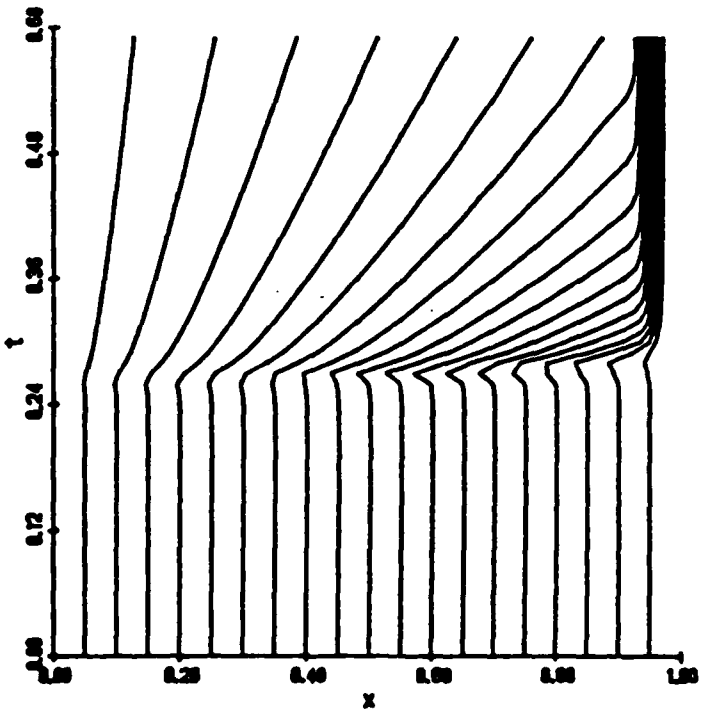


Figure 7. Mesh trajectories for Example 3, Eq. (4.8), with $\lambda = 10$ and $N = 20$

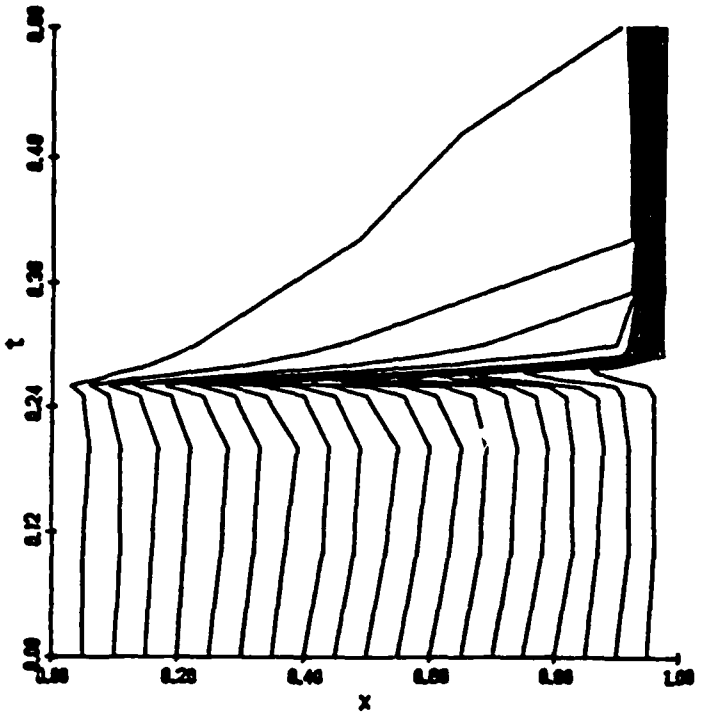


Figure 8. Mesh trajectories for Example 3, Eq. (4.8), with $\lambda = 200$ and $N = 20$.

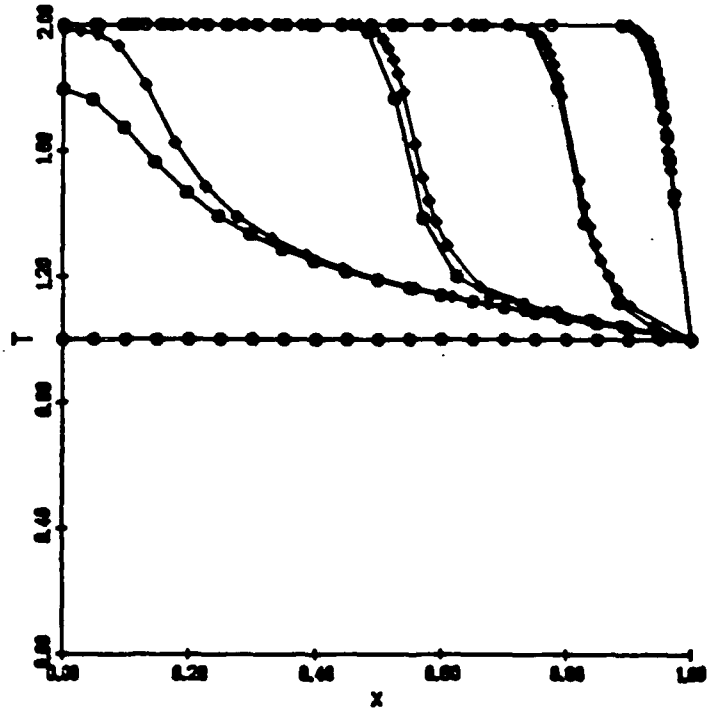


Figure 9. Temperature $T(x,t)$ vs. x at $t = 0.0$, 0.26, 0.27, 0.28 and $t = 0.29$ for Example 3, Eq. (4.8), with $N = 20$, $\lambda = 10$ (octagons), and $\lambda = 200$ (diamonds).

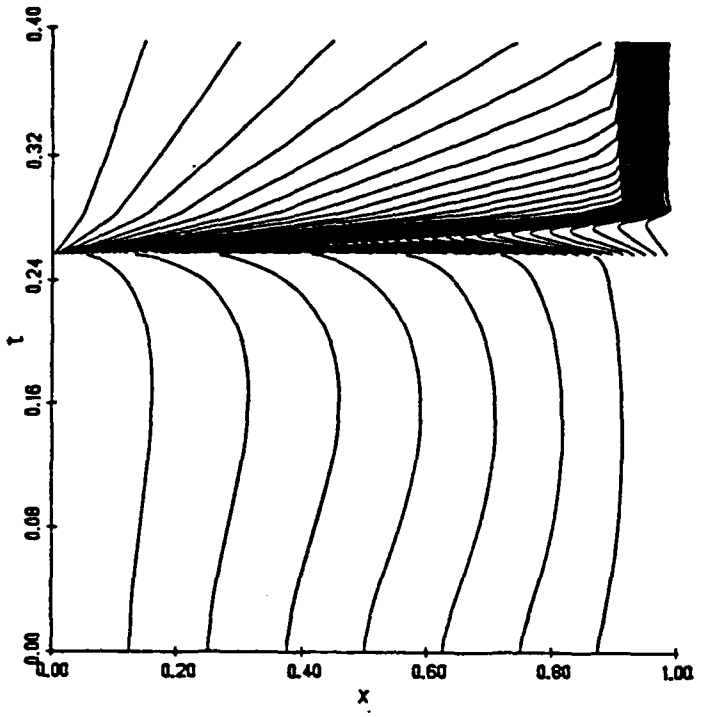


Figure 10. Mesh trajectories for Example 3, Eq. (4.8), using Refinement with tol = 0.1 and λ = 300.

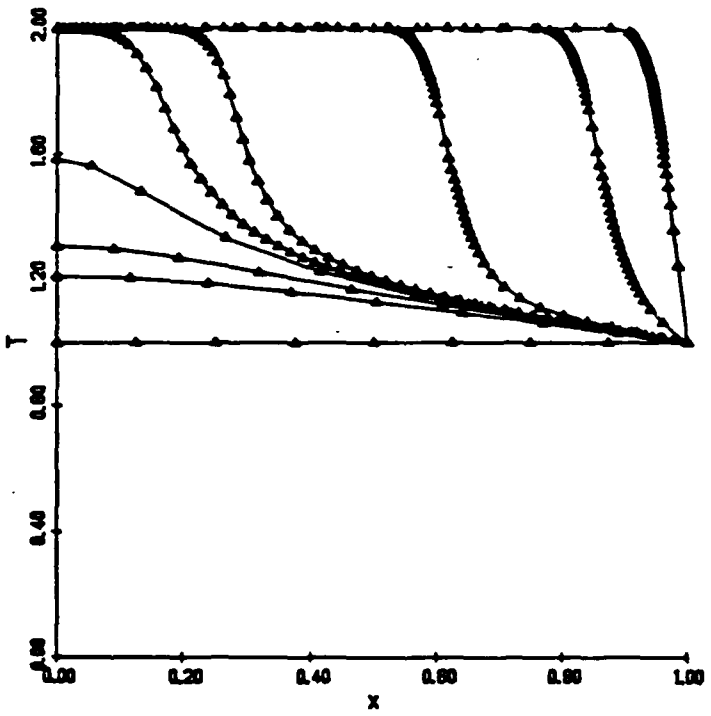


Figure 11. Temperature T(x,t) vs. x at t = 0.0, 0.24, 0.25, 0.256, 0.258, 0.26, 0.27, 0.28, and 0.6 for Example 3, Eq. (4.8), using Refinement with tol = 0.1 and λ = 300.

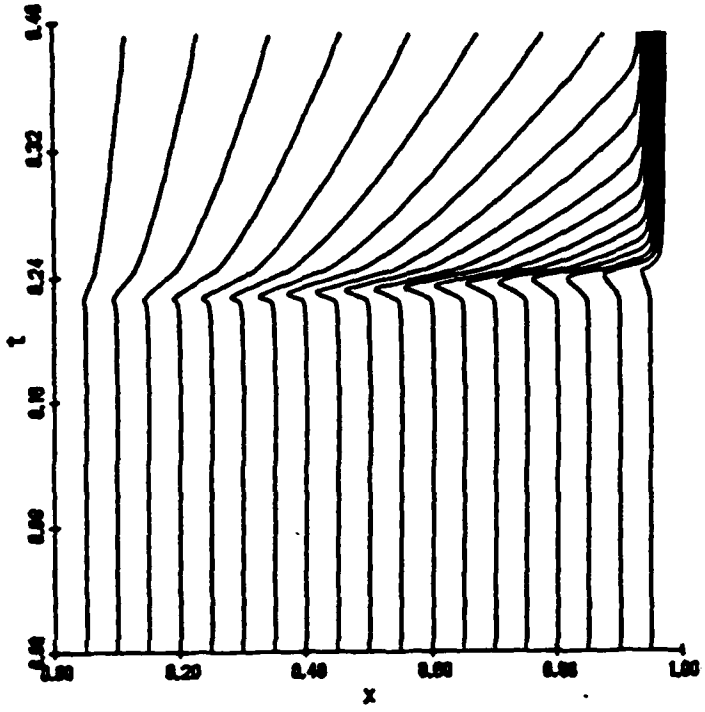


Figure 12. Mesh trajectories for Example 3, Eq. (4.7), with L = 0.9, λ = 10 and N = 20.

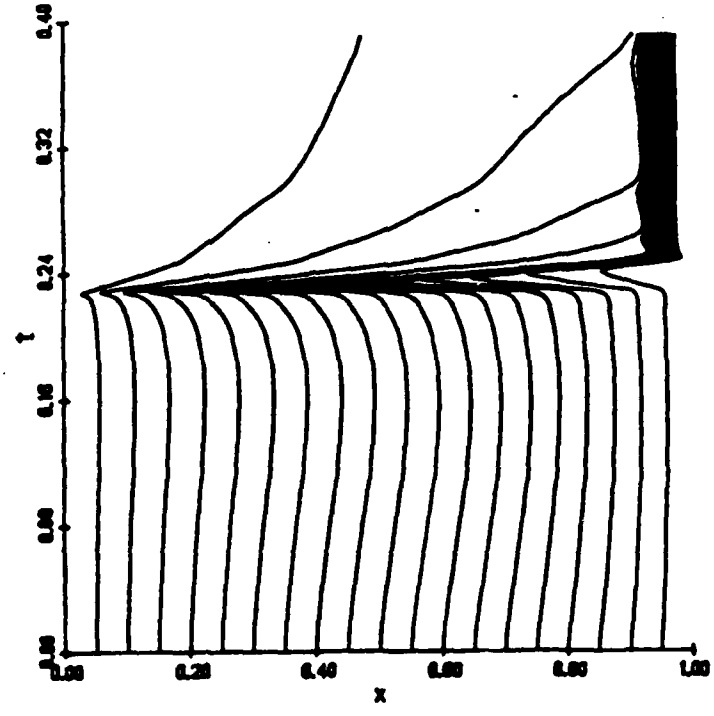


Figure 13. Mesh trajectories for Example 3, Eq. (4.7), with L = 0.9, λ = 150 and N = 20.

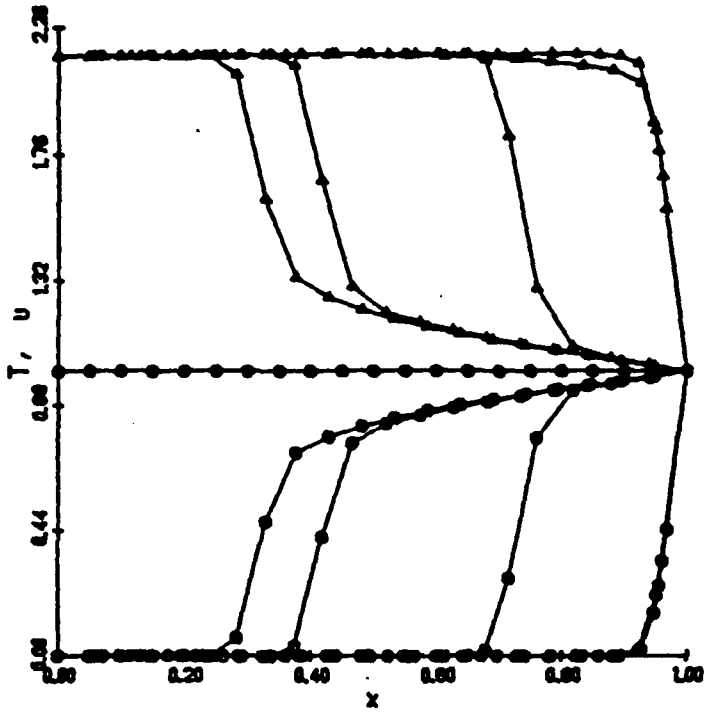


Figure 14. Temperature (triangles) and mass fraction (octagons) vs. x at t = 0.0, 0.227, 0.229, 0.237, 0.245, and 0.260 for Example 3, Eq. (4.7), with L = 0.9, λ = 10 and N = 20.

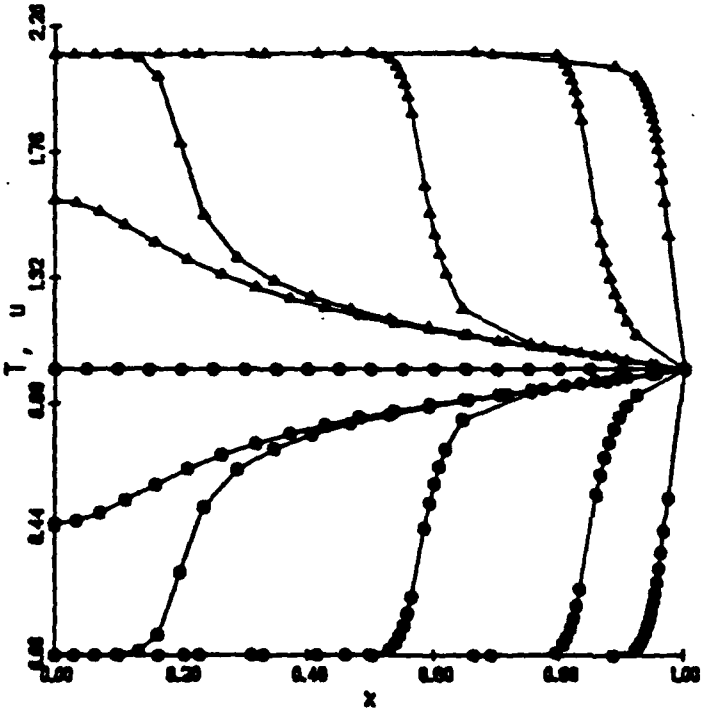


Figure 15. Temperature (triangles) and mass fraction (octagons) vs. x at t = 0.0, 0.227, 0.229, 0.237, 0.245, and 0.260 for Example 3, Eq. (4.7), with L = 0.9, λ = 150 and N = 20.

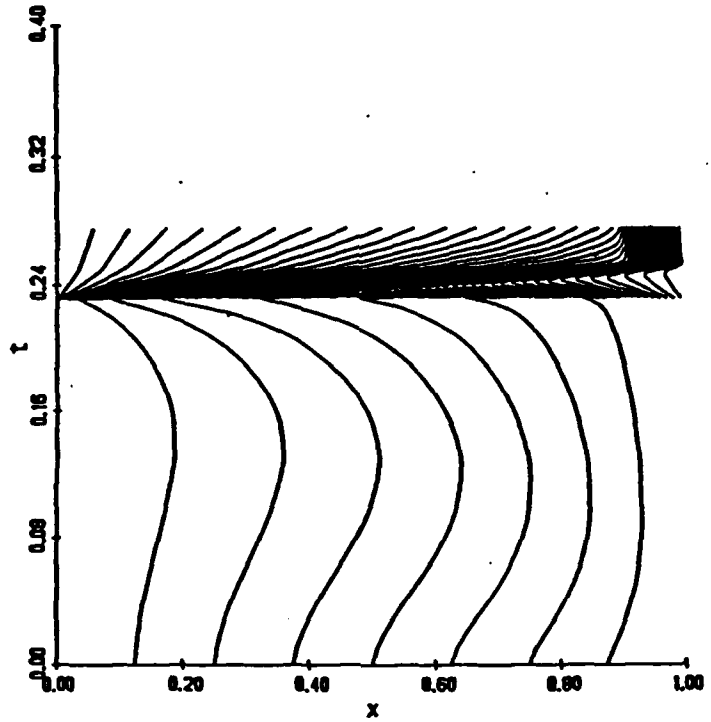


Figure 16. Mesh trajectories for Example 3, Eq. (4.7), using refinement with L = 0.9, λ = 500 and tol = 0.1.

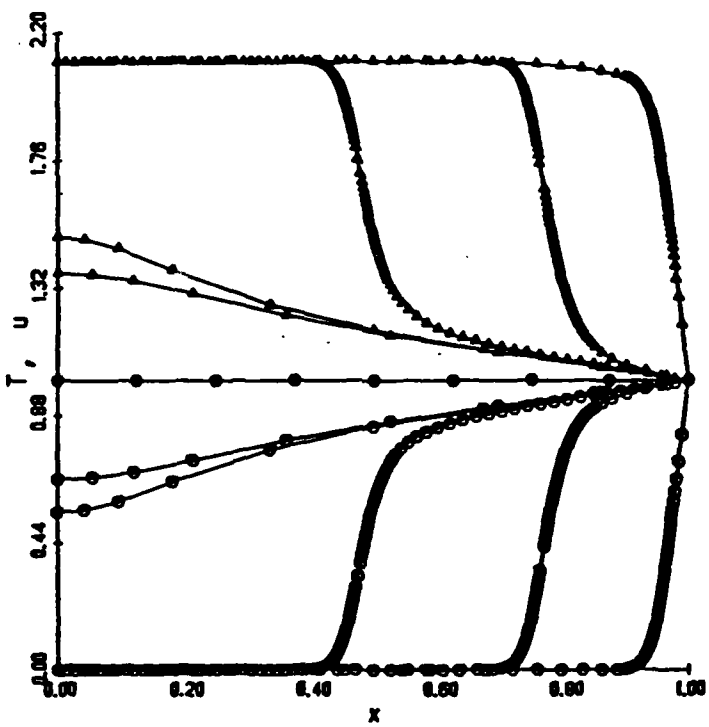


Figure 17. Temperature (triangles) and mass fraction (octagons) vs. x at t = 0.0, 0.227, 0.229, 0.237, 0.245, and 0.260 Eq. (4.7), using refinement with L = 0.9, λ = 500 and tol = 0.1.

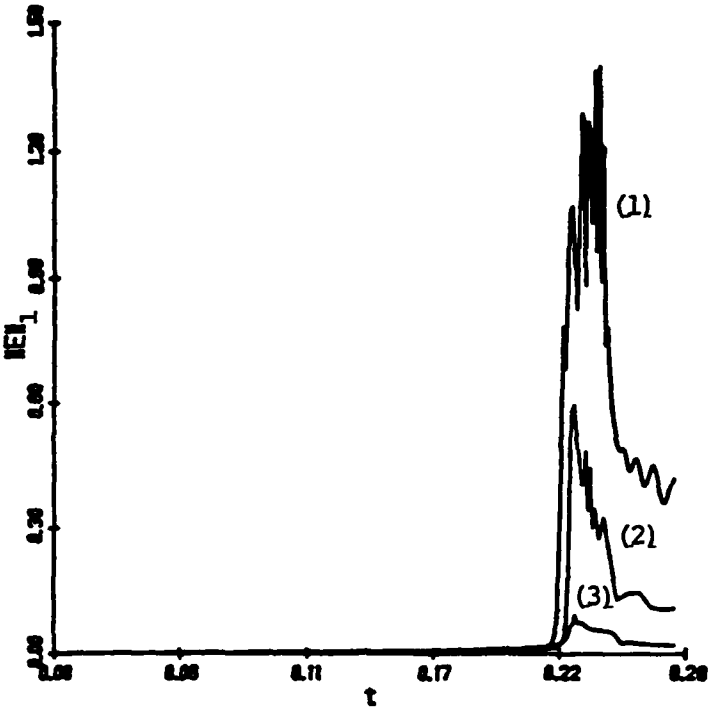
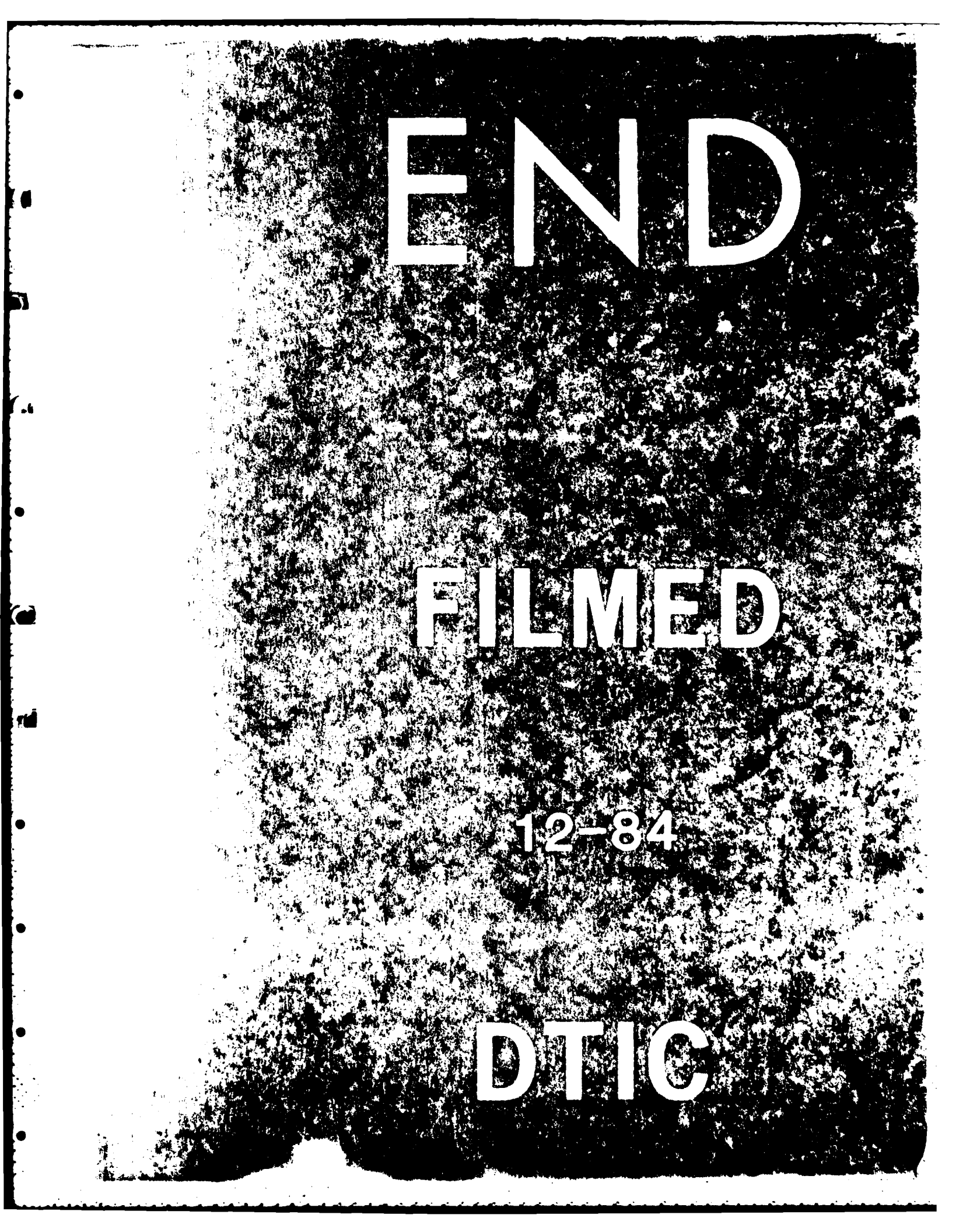


Figure 18. Estimated error $||E||_1$ vs. time for Example 3, eq. (4.7), with with L = 0.9, λ = 10, N = 20 (1), λ = 150, N = 20 (2), λ = 500 with refinement using tol = 0.1 (3).


END
FILMED
12-84
DTIC